

Appln. No. 09/834,851  
Amdt. dated August 12, 2005  
Reply to Office Action of May 9, 2005

PATENT

### REMARKS/ARGUMENTS

Claims 1-21 are pending and were variously rejected under 35 USC §102(e) as being anticipated by Barnett in view of Official Notice. In light of the remarks below, the undersigned respectfully traverses the rejections.

#### I. INITIAL MATTERS

Claims 1-21 were also provisionally rejected under the judicially created doctrine of obviousness-type double patenting over claims 1-21 of copending Application No. 09/834,855.

The Abstract was object to as being in claim format. It is not understood what language the Examiner objects to. The Abstract does not include any legal phraseology such as "means" or "said," but uses ordinary language, although the Abstract tracks the claims. Accordingly, the Undersigned does not understand the objection. Nevertheless, the undersigned notes that the original Abstract has greater than 150 words, accordingly, a new Abstract is provided.

In response, the undersigned respectfully requests that this provisional rejection be held in abeyance. If either or both of the copending applications are issued as patents before the present application issues as a patent, the undersigned is prepared to provide a terminal disclaimer in response to a non-provisional double patenting rejection.

Various amendments were also made to the claims to more clearly recite Markush-type claims. Such amendments were not made for purposes of patentability.

#### II. THE PRESENT INVENTION

The present invention relates to methods and systems for specifying promotions and distributing promotions across a computer network relying upon a unique and novel software architecture and mechanisms.

Initially, the specification distinguishes "promotions" or "electronic incentives" used herein from conventional "coupons." As described in the specification, page 15, lines 3-7:

*These promotions are not considered "coupons" as "coupons" is understood in the industry. More specifically, in the industry, "coupons" are typically defined as detachable certificates, tickets, or the like that entitle the bearer or holder to a benefit. In the present embodiment, the customer and the merchant server are not given any such detachable and/or possessable certificate and cannot hold, bear, or present anything.*

Additionally, the specification notes that coupons require possession of a cookie or the like:

*By way of contrast, in one electronic couponing systems, a electronic coupon describing a right or benefit is created in a couponing server. The electronic*

Appln. No. 09/834,851  
Amdt. dated August 12, 2005  
Reply to Office Action of May 9, 2005

PATENT

*coupon, or token, is then downloaded to a customer's computer system and stored. These coupons or tokens may be in the form of a cookie or the like stored on the customer's computer system. Much later, the customer may enter an electronic store that is independent of the electronic couponing system. Next, the cookie or token stored on the customer's computer system is retrieved and passed back to the electronic store web server. Because the customer's computer had "possession" of the cookie or token in the computer memory, the electronic store web server provides the customer the right or benefit or the bargain described, i.e. the customer is entitled to a 10% discount. This example thus illustrates that the electronic cookie or token incorporates the standard "coupon" model: the customer's computer memory stored the cookie, and possession of the cookie was a condition for receiving the bargain.*

The background of the invention describes some problems with these possessable coupons. More specifically, one problem is that coupons for a product may be provided to users who were already going to buy the product, p2, lines 1-12:

*A problem with traditional coupons includes that coupons often end up in the hands of buyers who are not targeted. This is because distributing coupons only to target buyers is virtually impossible. Although some coupons may be distributed to channels such as magazines, direct mailings, and the like that include a large percentage of target buyers, a significant percentage nevertheless reaches non-target buyers. These non-target buyers may include those willing to purchase the product even without the coupon. Accordingly, if non-target buyers uses the coupons to purchase a product, this directly reduces the amount of profit to the promoter. As an example, a promoter may create a promotion directed to Pepsi™ drinkers to try Coke™. To do so, the promoter offers coupons providing the bearer with a dollar off a six-pack of Coke™. However, it is virtually impossible to prevent a devoted Coke™ drinker from picking and redeem that coupon. This sort of common situation directly "siphons-off" manufacturer profits.*

In light of this problem, the specification states that improved apparatus for providing targeted promotions are needed, without the problems highlighted above.

Many of the amendments to the claims and distinctions over the cited art depend upon an understanding of the following specific software concepts: As expressly described in the specification, "object-oriented" software programming techniques are used, p. 14, lines 5-9, such as Microsoft COM software objects. For example, service objects, coupon objects, product objects, are described and used.

The specification should be read and claims should be interpreted in light of the object-oriented environment described. Particular terms related to object-oriented software were

Appln. No. 09/834,851  
Amdt. dated August 12, 2005  
Reply to Office Action of May 9, 2005

PATENT

defined and / or used in the specification consistently with how these terms are used in the software industry. The definitions of such terms in the software industry may override non-technical dictionary definitions of such words. For the Examiner's reference, particular definitions of terms are reproduced from the Microsoft Press Computer Dictionary, second edition, 1994 in attachment A to this amendment: object-oriented programming, object, instance, instantiate, and class.

Discussion of specific embodiments will be described below:

On p. 16, lines 10-12, the specification describes the merchant server invoking a Service object:

*[T]he merchant server invokes a Service object within the application server to evaluate the customer's shopping category to determine if there [are] any coupons to display, step 560.*

On p. 16, lines 12-14, the specification describes the application server instantiating coupon objects:

*In response to the current shopping category, the application server determines whether any promotions are applicable and if so, one or more "Coupon Objects" are instantiated, step 570.*

On p. 16, lines 25-38, the specification describes the merchant server querying the instances of the coupon objects:

*Next, merchant server 140 queries one or more "Coupon Objects" that have been instantiated for a description of the pre-conditions and benefit, a[n] image of the product, and the like, step 620. In response, merchant sever 140 specifies the rendering of the promotion on an HTML page for display on the customer's display, step 630.*

On p. 17, lines 22-26, the specification describes the merchant server invoking another service object;

*When the consumer desires to checkout, merchant server 140 causes application server 180 to use the instances of "Coupon Objects" that were created, step 710. In particular, an evaluate method of a Service object is invoked, and the amount of savings is calculated. The savings is then retrieved by merchant server 140 and displayed to the consumer, step 715.*

Appln. No. 09/834,851  
Amdt. dated August 12, 2005  
Reply to Office Action of May 9, 2005

PATENT

*In the present embodiment, when the consumer checks out, a promotion usage condition, application server 180 stores data associated with the transaction, step 720.*

The claims, as amended, incorporate at least some of the object-oriented concepts discussed above. For example, claim 1, now recites:

*wherein the instance of the electronic incentive is created and stored in the application server in response a method on a service object stored in the application server being invoked by the merchant server;*

*wherein the merchant server specifies rendering of the data associated with the electronic incentive in response to a query of the instance of the electronic incentive stored on the application server.*

For example, claim 8, now recites:

*a processor configured to request promotions from an application server coupled to the merchant server, configured to invoke an evaluation service object within an application server for one or more promotions, wherein an instance of a promotion is created in the application server in response thereto, configured to query the instance of the promotion object and receiving a description of a promotion from the application server, the description including pre-conditions, a user benefit and an output representation of the promotion, configured to transmit the output representation of the promotion to a client system for display to a user, configured to receive a selection of the at least one item, configured to invoke a savings method in a service object within the application server to determine a savings amount, wherein the savings amount comprises the user benefit from the application server when the selection of the at least one item satisfies the pre-conditions, and configured to indicate that the user is provided with the user benefit.*

For example, claim 15, now recites:

*a processor configured to receive an electronic incentive from a central server, the electronic incentive including a pre-condition and a benefit, configured to create an instance of the electronic incentive in response to an invocation of an evaluation service object to determine electronic incentives for a user by a merchant server, configured to receive a query for a description of the instance of the electronic incentive from the merchant server, configured to receive from the merchant server an invocation of an amount of savings method of a service object to determine a savings for the user, wherein when a selection by a user of at least one item fulfills the pre-condition of the electronic incentive, the savings comprises the benefit.*

Appln. No. 09/834,851  
Amdt. dated August 12, 2005  
Reply to Office Action of May 9, 2005

PATENT

### III. BARNETT

Barnett is described as a method and system for the electronic distribution of coupons to consumers. Specifically Barnett describes methods and systems where coupons bundles are provided to consumers via service providers.

Importantly Barnett does not refer to using an object-oriented paradigm. Instead, Barnett appears to simply rely upon simple procedural calls.

Additionally, Barnett appears to only refer to providing and redeeming conventional "coupons." Barnett, Fig. 9 includes a sample flow chart. In one step, the remote computer receives and stores variable "coupon data." Next, the coupon data is printed out and redeemed in-person, or the coupon is electronically redeemed. More specifically, the specification states on col. 9, lines 41-45.:

*The requested coupon data package and associated advertising materials are transmitted by the online service provider 2 to the personal computer 6, where it is stored in the downloaded coupon data file 30a in the coupon database.*

Next, the user prints out the coupons for redemption, col. 10 lines 58-60: Coupons are printed by the printable coupon data generation routine 32d, which is invoked by a user when he selects a print command from the coupon file function 56.

In the case of electronic redemption, the coupon is electronically transferred, col. 1, lines 38-44.:

*This is especially useful in the "electronic shopping mall" environment now found in many online services. The electronic coupon data could also be routed via the data communications interface 20 to a retail store where the user will be shopping, where the coupon data is held in a buffer pending purchase by the user of the matching product.*

To address the problem of unauthorized use or duplication of these coupons, Barnett describes using user-specific data in a bar code 90. Col. 7, line 24-25. Further, Barnett describes:

*The unique user bar code 90 also renders the electronic coupon system of the present invention secure and virtually fraud-proof. Although a user is able to print out a particular coupon 18 only once (to be described in detail below), the coupon issuer 14 could still be defrauded by a user or retailer who might photocopy a printed coupon numerous times and fraudulently and repeatedly present it for redemption. However, in accordance with the present invention, each coupon printed by a user is unique, and the scanning of a coupon presented for redemption will be*

Appln. No. 09/834,851  
Amdt. dated August 12, 2005  
Reply to Office Action of May 9, 2005

PATENT

*stored at the coupon redemption center. Thus, the coupon issuer will know if a particular user has redeemed a particular coupon and thus disallow further redemption of a photocopied coupon bearing the same indicia.*

However, in Barnett, because a user possesses these coupons, a user may still print-out a coupon for a product and give it to another user, who would have purchased the product even without the coupon. Accordingly, the manufacturer's profits may still undesirably be "siphoned-off" by these actions.

#### IV. BARNETT DISTINGUISHED

##### A. Claim 1

Barnett fails to disclose every element of claim 1.

More particularly, Barnett fails to disclose wherein the instance of the electronic incentive is created and stored in the application server in response to a method on a service object stored in the application server being invoked by the merchant server.

Further, Barnett fails to disclose wherein the merchant server specifies rendering of the data associated with the electronic incentive in response to a query of the instance of the electronic incentive stored on the application server.

As discussed above, Barnett fails to disclose anything about an implementation using an object-oriented approach and / or objects. Instead, Barnett simply describes that coupon data are simply downloaded from a online service provider to a user at a personal computer. Once the coupon data is on the personal computer in Barnett, the on line service provider loses control of the coupon.

In contrast, the claimed limitations describe the merchant server querying invoking methods of service objects and querying instances of electronic incentive objects stored on the application server. Additionally, the instance of the electronic incentive object on the application server provides providing data to end users via the merchant server, when needed. Accordingly, the electronic incentive is never downloaded as it is described in Barnett.

In light of the above, and for other reasons, Barnett fails to disclose all elements of claim 1. Accordingly, Barnett does not anticipate claim 1.

##### B. Claim 8

Barnett fails to disclose every element of claim 8. More specifically, Barnett fails to disclose the limitation of a processor configured to request promotions from an application server coupled to the merchant server, configured to invoke an evaluation service object within an application server for one or more promotions, wherein an instance of a promotion is created in the application server in response thereto, configured to query the instance of the promotion object and receiving a description of a promotion from the application server, the description including pre-conditions, a user benefit and an output representation of the promotion ,

Appln. No. 09/834,851  
Amdt. dated August 12, 2005  
Reply to Office Action of May 9, 2005

PATENT

configured to transmit the output representation of the promotion to a client system for display to a user, configured to receive a selection of the at least one item, configured to invoke a savings method in a service object within the application server to determine a savings amount, wherein the savings amount comprises the user benefit from the application server when the selection of the at least one item satisfies the pre-conditions, and configured to indicate that the user is provided with the user benefit

As discussed above, Barnett fails to disclose anything about an implementation using an object-oriented approach and /or objects. Additionally, Barnett simply describes that coupon data are simply requested and downloaded from a online service provider to a user at a personal computer.

In contrast, the claim language above illustrates the object-oriented aspect of embodiments of the present invention, which were not disclosed by Barnett, as well as the specific storage and invocations of methods of instances of software objects stored within the application server, and not within the merchant server.

In light of the above, and for other reasons, Barnett fails to disclose all elements of claim 8. Accordingly, Barnett does not anticipate claim 8.

C. Claim 15

Barnett fails to disclose every element of claim 15. More specifically Barnett fails to disclose a processor configured to receive an electronic incentive from a central server, the electronic incentive including a pre-condition and a benefit, configured to create an instance of the electronic incentive in response to an invocation of an evaluation service object to determine electronic incentives for a user by a merchant server, configured to receive a query for a description of the instance of the electronic incentive from the merchant server, configured to receive from the merchant server an invocation of an amount of savings method of a service object to determine a savings for the user, wherein when a selection by a user of at least one item fulfills the pre-condition of the electronic incentive, the savings comprises the benefit.

As summarized above, Barnett fails to disclose anything about an implementation using an object-oriented approach and /or objects. Instead, Barnett simply describes that coupon data are simply downloaded from a online service provider to a user at a personal computer. Because the coupon data is downloaded to the user's computer for the user to print out and / or use.

In contrast, the claim language above illustrates the object-oriented nature of embodiments of the present invention, which were not disclosed by Barnett, as well as the specific storage of the promotion object within the application server.

In light of the above, and for other reasons, Barnett fails to disclose all elements of claim 15. Accordingly, Barnett does not anticipate claim 15.

Appl. No. 09/834,851  
Amdt. dated August 12, 2005  
Reply to Office Action of May 9, 2005

PATENT

D. Remaining claims

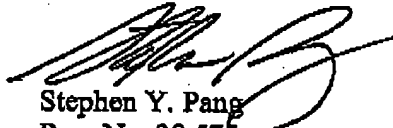
Claims 2-7; 9-14; 16-21, dependent upon claims 1, 8, and 15, respectively, are also asserted to be allowable for substantially the same reasons as claims 1, 8, and 15, respectively, and more specifically for the specific limitation they recite.

CONCLUSION

In view of the foregoing, Applicants believe all claims now pending in this Application are in condition for allowance. The issuance of a formal Notice of Allowance at an early date is respectfully requested.

If the Examiner believes a telephone conference would expedite prosecution of this application, please telephone the undersigned at (650) 326-2400.

Respectfully submitted,

  
Stephen Y. Pang  
Reg. No. 38,575

TOWNSEND and TOWNSEND and CREW LLP  
Two Embarcadero Center, Eighth Floor  
San Francisco, California 94111-3834  
Tel: (650) 326-2400  
Fax: (650) 326-2422

SYP:deh

Attachment: Appendix "The Comprehensive Standard for Business, School, Library, and Home"

60499609 v1



USPTO  
TO: Central Fax COMPANY:  
AUG. 12. 2005 3:32PM

9/16/2005 10:59 AM PAGE 23/026 Fax Server

TTC-PA 650-326-2422

--- --- "NO. 096" P.22 - - -

**APPENDIX**

FORWARDED BY  
Microsoft Press  
A Division of Microsoft Corporation  
One Microsoft Way  
Redmond, Washington 98058-6399  
Copyright © 1994 by Microsoft Press  
All rights reserved. No part of the contents of this book may be reproduced or  
transmitted in any form or by any means without the written permission of the publisher.  
Library of Congress Cataloging-in-Publication Data  
Microsoft Press computer dictionary : the encyclopedia standard for  
business, school, library, and home / Microsoft Press. -- 2nd ed.  
p. cm.  
ISBN 1-55558-997-3  
1. Computers--Dictionaries. 2. Microcomputers--Dictionaries.  
I. Microsoft Press. II. Title. Computer dictionary.  
QA76.93.M64 1993  
004.03--dc20  
93-3826  
CIP  
Printed and bound in the United States of America.  
456789 KJAL 94765  
Distributed in the book trade in Canada by Microsoft of Canada, a division of Canada  
Publishing Corporation.  
Distributed in the book trade outside the United States and Canada by  
Penguin Books Ltd.  
Penguin Books Ltd., Harmondsworth, Middlesex, England  
Penguin Books Australia Ltd., Ringwood, Victoria, Australia  
Penguin Books N.Z. Ltd., 100-109 Victoria Road, Auckland 10, New Zealand  
Tables Cataloging in Publication Data is available.  
Project Editors: Casey D. Doyle  
Manuscript Editors: Althea Crago Smith  
Technical Editors: Mary DeLong, Jeff Currey, Paul McGehee, Jr., Jim Pichler, Seth McGeevey

MICROSOFT PRESS

SECOND EDITION



THE COMPREHENSIVE  
STANDARD FOR  
BUSINESS, SCHOOL,  
LIBRARY, AND HOME

Microsoft  
P R E S S

object-oriented programming

also



object-oriented programming. Abbreviated OOP.

A programming paradigm in which a program is viewed as a collection of objects that are self-contained collections of data structures and methods that interact with other objects. A class defines the data structures and methods of an object, an object is an instance of a class that can be used as a variable in a program. In some object-oriented languages, objects respond to messages, which are the principal means of communication. Other object-oriented languages retain the traditional procedure-call mechanism. See also C++, object, Object-Oriented.

Object. A type of text created by sharing a resource to multiple tables within a relational database. Object text is on the computer or printer. See also table, text.

Object-oriented character recognition. A technique from the Latin word, meaning eight, this base-8 number system, consisting of the digits 0 through 7. The octal system is used in programming as a compact means of representing binary numbers. Because octal consists of eight digits and because 3 bits can form any of eight different combinations, binary numbers are commonly divided into groups of 3 bits for conversion to octal. For example, the binary equivalents of the eight octal digits are as follows:

Binary	Octal
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Thus the binary number 01100011 can be divided into groups of 3 bits, starting from the right and adding an extra 0 at the left, as 001 010 011. Compared to octal, the number becomes 123. Although, as this example shows, octal numbers can look like decimal numbers, their values differ because of the different meanings assigned to each number position. In decimal notation, for

277

ject. An informal term for object code that also means machine code, in object-oriented programming, a variable containing both routines and data that is treated as a discrete entity. See also abstract data type, module, object code, object-oriented programming.

term or language that supports the use of objects. See also object.

Object-oriented graphics. Also called structured graphics. Computer graphics that use based on the use of "primitive elements" (graphics primitives), such as lines, curves, circles, and squares. Object-oriented graphics, used in applications such as computer-aided design and drawing and illustration programs, describe an image mathematically as a set of instructions for creating the objects in the image. This approach contrasts with bit-mapped graphics, the other widely used approach to creating images, which represents a graphic as a group of black and white or colored dots arranged in a certain pattern. Object-oriented graphics enable the user to manipulate objects as entire units—for example, to change the length of a line or enlarge a circle—whereas bit-mapped graphics require repainting individual data in the line or circle. Because objects are described mathematically, object-oriented graphics can also be layered, stored, and manipulated relatively easily. Computer-generated graphics can also be layered, stored, and manipulated. Object-oriented interfaces. A type of user interface in which elements of the system are represented by visible icons, buttons, and icons (quasi-representational), which are used to manipulate the system elements. For example, the Macintosh Finder presents an object-oriented interface to the file system, representing it by using images of documents, file folders, and disk drives. Object-oriented display interfaces do not necessarily imply any relation to object-oriented programming. See also object-oriented graphics.

276

object. An informal term for object code that also means machine code, in object-oriented programming, a variable containing both routines and data that is treated as a discrete entity. See also abstract data type, module, object code, object-oriented programming.

In graphics, a display entity. For example, a bounding box might be an object in a graphics program.

Object code. The code, generated by a compiler or an assembler, that was translated from the source code of a program. The term must commonly refer to machine code that can be directly executed by the system's central processing unit (CPU), but it can also be assembly language source code or a version of machine code. Compare source code; see also assembly language, compiler.

Object compiler. The compiler that is the target of a specific communications attempt.

Object file. A file containing object code, usually the output of a compiler or an assembler and the input for a linker. See also object code.

Objective-C. An object-oriented version of the C language developed in 1988 by Brad Cox. It is not widely known for being the standard development language for the NEXT system. See also object-oriented programming.

276

transmission program



transmission program

transmission program. A program whose function is to send another program, either on a storage medium or in memory. An instruction program might be used to guide a user through the often complex process of setting up an application for a particular combination of machine, printer, and number. Instruction programs are also used when an application is copy-protected and cannot be copied by normal operating-system commands. Such instruction programs typically limit the number of copies that can be made, remove a copy that has been installed on one machine to another machine, the user must download a copy and activate it on the other machine (often with the same installation program).

transmission program. A program provided by Apple with each new release of the Macintosh operating system. The loader allows the user to install system updates and to make bootable (system) disks. In the example, if you define a class called *File* and then create (allocate memory for) a *File* object called *myFile*, you've created an instance of the class *File*. See also *class*, *instance variable*, *instance variable*.

transmission program. An action taken in any computer language (machine, assembly, high-level), although more often used with reference to assembly language programs. Most programs can be broken down into two types of statements: instructions and declarations. See also *declaration*, *statement*.

transmission program. An action taken in any computer language (machine, assembly, high-level), although more often used with reference to assembly language programs. Most programs can be broken down into two types of statements: instructions and declarations. See also *declaration*, *statement*.

transmission program. An action taken in any computer language (machine, assembly, high-level), although more often used with reference to assembly language programs. Most programs can be broken down into two types of statements: instructions and declarations. See also *declaration*, *statement*.

215

transmission program

transmission program



transmission program

transmission program. A program whose function is to send another program, either on a storage medium or in memory. An instruction program might be used to guide a user through the often complex process of setting up an application for a particular combination of machine, printer, and number. Instruction programs are also used when an application is copy-protected and cannot be copied by normal operating-system commands. Such instruction programs typically limit the number of copies that can be made, remove a copy that has been installed on one machine to another machine, the user must download a copy and activate it on the other machine (often with the same installation program).

transmission program. A program provided by Apple with each new release of the Macintosh operating system. The loader allows the user to install system updates and to make bootable (system) disks. In the example, if you define a class called *File* and then create (allocate memory for) a *File* object called *myFile*, you've created an instance of the class *File*. See also *class*, *instance variable*, *instance variable*.

transmission program. An action taken in any computer language (machine, assembly, high-level), although more often used with reference to assembly language programs. Most programs can be broken down into two types of statements: instructions and declarations. See also *declaration*, *statement*.

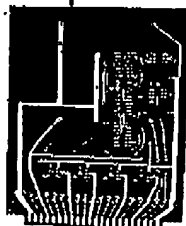
transmission program. An action taken in any computer language (machine, assembly, high-level), although more often used with reference to assembly language programs. Most programs can be broken down into two types of statements: instructions and declarations. See also *declaration*, *statement*.

74

transmission program. A program whose function is to send another program, either on a storage medium or in memory. An instruction program might be used to guide a user through the often complex process of setting up an application for a particular combination of machine, printer, and number. Instruction programs are also used when an application is copy-protected and cannot be copied by normal operating-system commands. Such instruction programs typically limit the number of copies that can be made, remove a copy that has been installed on one machine to another machine, the user must download a copy and activate it on the other machine (often with the same installation program).

transmission program. A program provided by Apple with each new release of the Macintosh operating system. The loader allows the user to install system updates and to make bootable (system) disks. In the example, if you define a class called *File* and then create (allocate memory for) a *File* object called *myFile*, you've created an instance of the class *File*. See also *class*, *instance variable*, *instance variable*.

transmission program. An action taken in any computer language (machine, assembly, high-level), although more often used with reference to assembly language programs. Most programs can be broken down into two types of statements: instructions and declarations. See also *declaration*, *statement*.



Circuit board

Circuit board. A switch that opens and cuts off the flow of current when the current exceeds a certain level. Circuit breakers are placed at critical points in circuits to protect against damage that could result from excessive current flow, which is typically caused by component failure. Circuit breakers are often used in place of fuses because they need only to be reset rather than replaced. Compare *overcurrent protection*.

Circuit board. A switch that opens and cuts off the flow of current when the current exceeds a certain level. Circuit breakers are placed at critical points in circuits to protect against damage that could result from excessive current flow, which is typically caused by component failure. Circuit breakers are often used in place of fuses because they need only to be reset rather than replaced. Compare *overcurrent protection*.